Bayesian Regression and Classification

Bayesian Regression and Classification

Christopher M. Bishop and Michael E. Tipping
Microsoft Research, 7 J J Thomson Avenue, Cambridge, CB3 0FB, U.K.

To appear in *Advances in Learning Theory: Methods, Models and Applications*, J.A.K. Suykens *et al.* (Editors), IOS Press, NATO Science Series III: Computer and Systems Sciences, volume **190**.

Abstract In recent years Bayesian methods have become widespread in many domains including computer vision, signal processing, information retrieval and genome data analysis. The availability of fast computers allows the required computations to be performed in reasonable time, and thereby makes the benefits of a Bayesian treatment accessible to an ever broadening range of applications. In this tutorial we give an overview of the Bayesian approach to pattern recognition in the context of simple regression and classification problems. We then describe in detail a specific Bayesian model for regression and classification called the *Relevance Vector Machine*. This overcomes many of the limitations of the widely used Support Vector Machine, while retaining the highly desirable property of sparseness.

1 Introduction

Although Bayesian methods have been studied for many years, it is only recently that their practical application has become truly widespread. This is due in large part to the relatively high computational overhead of performing the marginalizations (integrations and summations) which lie at the heart of the Bayesian paradigm. For this reason more traditional approaches, based on point estimation of parameters, have typically been the method of choice. However, the widespread availability of fast computers allows Bayesian computations to be performed in reasonable time for an increasingly wide spectrum of real world applications. Furthermore, the development of Markov chain Monte Carlo techniques, and more recently of deterministic approximation schemes such as variational inference, have greatly extended the range of models amenable to a Bayesian treatment.

1.1 Least Squares Regression

In this tutorial we consider the relatively simple, but widely studied, problems of regression and classification for independent, identically distributed (i.i.d.) data. Consider a data set of examples of input vectors $\{\mathbf{x}_n\}_{n=1}^N$ along with corresponding targets $\mathbf{t} = \{t_n\}_{n=1}^N$. Note that, for notational simplicity, we shall consider a single target variable, but that the extension of the methods discussed in this paper to multiple target variables is straightforward. For regression, we generally assume that the targets are some noisy realization of an underlying functional relationship $y(\mathbf{x})$ that we wish to estimate so that

$$t_n = y(\mathbf{x}_n; \mathbf{w}) + \epsilon_n \tag{1}$$

where ϵ is an additive noise process in which the values ϵ_n are i.i.d., and $\mathbf{w}$ is a vector of adjustable parameters or 'weights'.

One interesting class of candidate functions for $y(\mathbf{x}; \mathbf{w})$ is given by

$$y(\mathbf{x}; \mathbf{w}) = \sum_{i=1}^{M} w_i \phi_i(\mathbf{x}) = \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}), \tag{2}$$

which represents a linearly-weighted sum of M nonlinear fixed basis functions denoted by $\boldsymbol{\phi}(\mathbf{x}) = (\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots, \phi_M(\mathbf{x}))^{\mathrm{T}}$. Models the type (2) are known as *linear* models since the function $y(\mathbf{x}; \mathbf{w})$ is a linear function of the parameters $\mathbf{w} = (w_1, w_2, \ldots, w_M)^{\mathrm{T}}$. However, in general the function itself is non-linear, and indeed can be very flexible if M is relatively large.

Classical (non-Bayesian) techniques use some form of 'estimator' to determine a specific value for the parameter vector $\mathbf{w}$. One of the simplest examples is the sum-of-squares error function defined by

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} |y(\mathbf{x}_n; \mathbf{w}) - t_n|^2 \tag{3}$$

where the factor of $1/2$ is included for later convenience. Minimizing this error function with respect to $\mathbf{w}$ leads to an estimate $\mathbf{w}^*$ which can be used to make predictions for new values of $\mathbf{x}$ by evaluating $y(\mathbf{x}; \mathbf{w}^*)$.

In the case of classification problems, the function $y(\mathbf{x}; \mathbf{w})$ is transformed using an appropriate non-linearity, such as a logistic sigmoid for 2-class problems or a softmax (normalized exponential) for multi-class problems. The corresponding error function is given by the cross-entropy (Bishop 1995).

A well-known problem with error function minimization is that complex and flexible models can 'over-fit' the training data, leading to poor generalization. Indeed, when the number of parameters equals the number of data points, the least squares solution for a model of the form (2) can achieve a perfect fit to the training data while having very poor generalization to new data points. This behaviour is characterized by value of the parameters w_i which have large positive and negative values finely tuned to the individual noisy data points. The corresponding function $y(\mathbf{x}; \mathbf{w})$ typically exhibits strong oscillations as a function of $\mathbf{x}$. Whilst over-fitting can be avoided by limiting the complexity of the model, this too can lead to poor generalization if the model is insufficiently flexible to capture the underlying behaviour of the data set. However, we often have to work with data sets of limited size and yet we wish to be able to use flexible models many adjustable parameters. We shall see that the phenomenon of over-fitting is a pathological property of point estimation, and that by adopting a Bayesian viewpoint we can apply complex models to small data sets without encountering problems of over-fitting.

1.2 Regularization

One classical (non-Bayesian) technique for reducing over-fitting is that of regularization in which a penalty term $\Omega(\mathbf{w})$ is added to the error function to give

$$\widetilde{E}(\mathbf{w}) = E(\mathbf{w}) + \lambda \Omega(\mathbf{w}) \tag{4}$$

where $\Omega(\mathbf{w})$ discourages over-fitting, for example by penalizing large values for the weight parameters w_i. The parameter λ controls the trade-off between fitting the data by reducing $E(\mathbf{w})$ and smoothing the function $y(\mathbf{x};\mathbf{w})$ as a function of $\mathbf{x}$ by reducing $\Omega(\mathbf{w})$. A common choice of regularizer is given by the sum of the squares of the weight parameters, so that

$$\Omega(\mathbf{w}) = \frac{\|\mathbf{w}\|^2}{2}. \tag{5}$$

The value of the regularization coefficient is typically set by holding back some data from the training set and optimizing λ by minimizing the value of the un-regularized error function $E(\mathbf{w})$ evaluated with respect to the held out data. For small data sets this procedure may be refined to give the cross-validation technique which makes more efficient use of limited data (Bishop 1995).

1.3 Probabilistic Models

We can motivate the regularized least-squares framework from a probabilistic viewpoint as follows. The observed target values t_n are assumed to have been generated from the underlying function $y(\mathbf{x};\mathbf{w})$ by the addition of independent Gaussian noise, so that in (1) the noise values ϵ_n are normally distributed with zero mean and variance σ^2 so that

$$p(\epsilon|\sigma^2) = \mathcal{N}(\epsilon|0,\sigma^2) \tag{6}$$

$$= \left(\frac{1}{2\pi\sigma^2}\right)^{1/2} \exp\left\{-\frac{1}{2\sigma^2}\epsilon^2\right\} \tag{7}$$

where the notation $\mathcal{N}(\epsilon|\mu,\sigma^2)$ specifies a Gaussian distribution over ϵ with mean μ and variance σ^2. Variables such as μ and σ^2 are sometimes called *hyperparameters* since they control the distribution over parameters. From (1) and (7) it follows that the conditional distribution of the target variable given the input variable and the weight parameters is again a Gaussian distribution

$$p(t|\mathbf{x},\mathbf{w},\sigma^2) = \left(\frac{1}{2\pi\sigma^2}\right)^{1/2} \exp\left\{-\frac{1}{2\sigma^2}|y(\mathbf{x};\mathbf{w}) - t|^2\right\}. \tag{8}$$

Note that the distribution for t is conditioned on the value of $\mathbf{x}$. We are not interested in modelling the distribution of $\mathbf{x}$ and so from now on we shall omit $\mathbf{x}$ from the conditioning list in order to keep the notation compact. Since the data points are independent, the joint probability of the whole data set, given $\mathbf{w}$ and β, is given by the product over all data points of the conditional distribution (8) evaluated at the observed data values

$$L(\mathbf{w}) = p(\mathbf{t}|\mathbf{w},\sigma^2) = \left(\frac{1}{2\pi\sigma^2}\right)^{N/2} \exp\left\{-\frac{1}{2\sigma^2}\sum_{n=1}^{N}|y(\mathbf{x}_n;\mathbf{w}) - t_n|^2\right\}. \tag{9}$$

When viewed as a function of $\mathbf{w}$ this is called the *likelihood function*.

One technique from classical statistics for estimating $\mathbf{w}$ is called *maximum likelihood* and involves setting $\mathbf{w}$ to the value which maximizes the likelihood function. For convenience we can instead minimize the negative logarithm of the likelihood function (since '$-\ln$' is a monotonically decreasing function) given by

$$-\ln L(\mathbf{w}) = \frac{N}{2}\ln\sigma^2 + \frac{N}{2}\ln(2\pi) + \frac{1}{2\sigma^2}\sum_{n=1}^{N}|y(\mathbf{x}_n;\mathbf{w}) - t_n|^2. \tag{10}$$

Note that minimizing $\ln L(\mathbf{w})$ in (10) with respect to $\mathbf{w}$ is equivalent to minimizing the sum of squares error function (3). We denote the resulting value of $\mathbf{w}$ by $\mathbf{w}_{\mathrm{ML}}$. Similarly we can minimize (10) with respect to β with the result

$$\frac{1}{\beta_{\mathrm{ML}}} = \frac{1}{N}\sum_{n=1}^{N} |y(\mathbf{x}_n; \mathbf{w}_{\mathrm{ML}}) - t_n|^2. \tag{11}$$

This provides us with an estimate of the noise level associated with the data under the assumed model.

1.4 Bayesian Regression

We have seen that a classical treatment of our regression problem seeks a point estimate of the unknown parameter vector $\mathbf{w}$. By contrast, in a Bayesian approach we characterize the uncertainty in $\mathbf{w}$ through a probability distribution $p(\mathbf{w})$. Observations of data points modify this distribution by virtue of Bayes' theorem, with the effect of the data being mediated through the likelihood function.

Specifically we define a prior distribution $p(\mathbf{w})$ which expresses our uncertainty in $\mathbf{w}$ taking account of all information aside from the data itself, and which, without loss of generality, can be written in the form

$$p(\mathbf{w}|\alpha) \propto \exp\{-\alpha\Omega(\mathbf{w})\} \tag{12}$$

where α can again be regarded as a hyperparameter. As a specific example we might choose a Gaussian distribution for $p(\mathbf{w}|\alpha)$ of the form

$$p(\mathbf{w}|\alpha) = \left(\frac{\alpha}{2\pi}\right)^{M/2} \exp\left\{-\frac{\alpha}{2}\|\mathbf{w}\|^2\right\}. \tag{13}$$

We can now use Bayes' theorem to express the posterior distribution for $\mathbf{w}$ as the product of the prior distribution and the likelihood function

$$p(\mathbf{w}|\mathbf{t}, \alpha, \sigma^2) \propto p(\mathbf{w}|\alpha)L(\mathbf{w}) \tag{14}$$

where, as before, $L(\mathbf{w}) = p(\mathbf{t}|\mathbf{w}, \sigma^2)$.

In a Bayesian treatment we make predictions by integrating with respect to the posterior distribution of $\mathbf{w}$, and we discuss this in detail shortly. For the moment, let us suppose that we wish to use the posterior distribution to find a point estimate for $\mathbf{w}$, and that we choose to do this by finding the value of $\mathbf{w}$ which maximizes the posterior distribution, or equivalently which minimizes the negative logarithm of the distribution. Taking the negative log of the right hand side of (14) and using (12) and (9) we see that maximizing the log of the posterior distribution is equivalent to minimizing

$$\frac{1}{2\sigma^2}\sum_{n=1}^{N} |y(\mathbf{x}_n; \mathbf{w}) - t_n|^2 + \frac{\alpha}{2}\Omega(\mathbf{w}) \tag{15}$$

which represents a specific example of the regularized error function given by (4) in which $E(\mathbf{w})$ is proportional to the sum-of-squares error function (3).

Thus we see that there are very close similarities between this Bayesian viewpoint and the conventional one based on error function minimization and regularization, since the latter can

be obtained as a specific approximation to the Bayesian approach. However, there is also a key distinction which is that in a Bayesian treatment we make predictions by *integrating* over the distribution of model parameters $\mathbf{w}$, rather than by using a specific estimated value of $\mathbf{w}$. On the one hand such integrations may often be analytically intractable and require either sophisticated Markov chain Monte Carlo methods, or more recent deterministic schemes such as variational techniques, to approximate them. On the other hand the integration implied by the Bayesian framework overcomes the issue of over-fitting (by averaging over many different possible solutions) and typically results in improved predictive capability.

Specifically, if we are given a new value of $\mathbf{x}$ then the predictive distribution for t is obtained from the sum and product rules of probability by marginalizing over $\mathbf{w}$

$$p(t|\mathbf{t}, \alpha, \beta) = \int p(\mathbf{w}|\mathbf{t}, \alpha, \sigma^2) p(t|\mathbf{w}, \sigma^2)\, d\mathbf{w}. \tag{16}$$

So far we have said little about the treatment of the hyperparameters α and σ^2. In most applications, suitable values for these will not be known in advance (although in some cases the noise level σ^2 may be known) and so a Bayesian treatment will introduce prior distributions over these quantities, and then eliminate them from the problem by marginalization. We shall see in Section 3 that an appropriate choice of prior distribution can lead to some powerful properties for the resulting model, including sparsity of the basis function representation. First, however, we review briefly a popular model for regression and classification based on point estimates of parameters, which also exhibits sparsity.

2 Support Vector Machines

One specific instantiation of the model given by (2) is the *support vector machine* (SVM) (Boser, Guyon, and Vapnik 1992; Vapnik 1998; Schölkopf, Burges, and Smola 1999) which, although not usually defined explicitly in this form, ultimately makes predictions based on the function

$$y(\mathbf{x}; \mathbf{w}) = \sum_{i=1}^{N} w_i K(\mathbf{x}, \mathbf{x}_i) + w_0. \tag{17}$$

Here $\phi_i(\mathbf{x}) = K(\mathbf{x}, \mathbf{x}_i)$ is a *kernel* function, effectively defining one basis function for each example in the training set. The key feature of the SVM is that, in the classification case, its target function attempts to minimize a measure of error on the training set while simultaneously maximizing the 'margin' between the two classes (in the feature space implicitly defined by the kernel). This is an effective mechanism for avoiding over-fitting, which leads to good generalization, and which furthermore results in a sparse model dependent only on a subset of kernel functions, namely those associated with specific training examples $\mathbf{x}_n$ (the *support vectors*) that lie either on the margin or on the 'wrong' side of it. State-of-the-art results have been reported on many tasks where the SVM has been applied.

However, despite its success, we can identify a number of significant and practical disadvantages of the support vector learning methodology:

- SVMs make unnecessarily liberal use of basis functions since the number of support vectors required typically grows linearly with the size of the training set.

- Predictions are not *probabilistic*. In regression the SVM outputs a point estimate, and in classification, a 'hard' binary decision. For many real world applications, as distinct from algorithm bench-marking, we require the conditional distribution $p(t|\mathbf{x})$ of targets given inputs rather than just a point prediction. Such a distribution expresses our uncertainty in the prediction and offers numerous advantages (Bishop 1995) such as optimal rejection, flexible and optimal decision making, fusion of outputs with other sources of probabilistic information, and so on.
- It is necessary to estimate the error/margin trade-off parameter 'C' (and in regression, the insensitivity parameter 'ϵ' too). This generally entails a cross-validation procedure, which is wasteful both of data and computation.
- The kernel function $K(\mathbf{x}, \mathbf{x}_i)$ must satisfy Mercer's condition.

Nevertheless, the twin properties of accuracy and sparsity make the SVM a very attractive model. We have already discussed how a Bayesian approach to modelling can naturally deal with complexity control and avoid over-fitting. Here we show that in addition, through a judicious choice of prior over $\mathbf{w}$, we can obtain models that are also highly sparse (typically much more so than the SVM) and at the same time also overcome all the above limitations.

3 The Relevance Vector Machine

While we stress that the framework we are about to describe can be applied to general models of the type (2) (*i.e.* to arbitrary sets of basis functions), we now focus on a model we term the *relevance vector machine*, or RVM (Tipping 2000; Tipping 2001), which is a Bayesian framework for regression and classification with analogous sparsity properties to the support vector machine. We adopt a fully probabilistic framework and introduce a prior over the model weights governed by a set of hyperparameters, one associated with each weight, whose most probable values are iteratively estimated from the data. Sparsity is achieved because the posterior distributions of many of the weights are sharply (indeed infinitely) peaked around zero. We term those training vectors associated with the remaining non-zero weights 'relevance' vectors, in deference to the principle of *automatic relevance determination* which motivates this approach (MacKay 1994; Neal 1996). The most compelling feature of the RVM is that, while capable of generalization performance comparable to an equivalent SVM, the number of relevance vectors is, in most cases, dramatically smaller than the number of support vectors used by an SVM to solve the same problem. For the purposes of this presentation, we focus initially on the Bayesian *regression* model and associated inference procedures, and then summarize the modifications required in the case of classification.

3.1 Model Specification

Given a data set of input-target pairs $\{\mathbf{x}_n, t_n\}_{n=1}^N$ we assume that the targets are samples from a model with additive noise, as described by (1), with a noise process given by a zero-mean Gaussian with variance σ^2, so that

$$p(t_n|\mathbf{x}) = \mathcal{N}(t_n|y(\mathbf{x}_n; \mathbf{w}), \sigma^2). \tag{18}$$

The function $y(\mathbf{x};\mathbf{w})$ is as defined in (17) for the SVM where we identify our general basis functions with the kernel as parameterized by the training vectors: $\phi_i(\mathbf{x}) \equiv K(\mathbf{x},\mathbf{x}_i)$. Due to the assumption of independence of the t_n, the likelihood of the complete data set can be written as

$$p(\mathbf{t}|\mathbf{w},\sigma^2) = (2\pi\sigma^2)^{-N/2}\exp\left\{-\frac{1}{2\sigma^2}\|\mathbf{t}-\mathbf{\Phi}\mathbf{w}\|^2\right\}, \tag{19}$$

where the $N\times(N+1)$ matrix $\mathbf{\Phi} = [\boldsymbol{\phi}(\mathbf{x}_1),\boldsymbol{\phi}(\mathbf{x}_2),\ldots,\boldsymbol{\phi}(\mathbf{x}_N)]^{\mathrm{T}}$ is called the *design* matrix, $\boldsymbol{\phi}(\mathbf{x}_n) = [1, K(\mathbf{x}_n,\mathbf{x}_1), K(\mathbf{x}_n,\mathbf{x}_2),\ldots,K(\mathbf{x}_n,\mathbf{x}_N)]^{\mathrm{T}}$, $\mathbf{t} = (t_1\ldots t_N)^{\mathrm{T}}$, and $\mathbf{w} = (w_0\ldots w_N)^{\mathrm{T}}$.

With as many parameters in the model as training examples, we would expect maximum-likelihood estimation of $\mathbf{w}$ and σ^2 from (19) to lead to severe over-fitting. In the SVM, this difficulty is effectively avoided by the inclusion of the 'margin' term. Here, instead, we adopt a Bayesian perspective, and introduce an explicit *prior* probability distribution over the parameters.

We encode a preference for smoother functions by using a Gaussian prior distribution over $\mathbf{w}$, as discussed earlier, but now modified through the introduction of a separate hyperparameter for each parameter in the model

$$p(\mathbf{w}|\boldsymbol{\alpha}) = \prod_{i=0}^{N}\mathcal{N}(w_i|0,\alpha_i^{-1}), \tag{20}$$

with $\boldsymbol{\alpha}$ a vector of $N+1$ hyperparameters.

To complete the specification of this *hierarchical* prior, we must define hyperpriors over $\boldsymbol{\alpha}$, as well as over the final remaining parameter in the model, the noise variance σ^2. These quantities are examples of *scale* parameters, and suitable priors for these are given by Gamma distributions (see, *e.g.* Berger (1985)):

$$p(\boldsymbol{\alpha}) = \prod_{i=0}^{N}\text{Gamma}(\alpha_i|a,b),$$
$$p(\beta) = \text{Gamma}(\beta|c,d),$$

with $\beta \equiv \sigma^{-2}$ and where

$$\text{Gamma}(\alpha|a,b) = \Gamma(a)^{-1}b^a\alpha^{a-1}e^{-b\alpha}, \tag{21}$$

in which $\Gamma(a) = \int_0^\infty t^{a-1}e^{-t}dt$, is the gamma function (Abromowitz and Stegun 1965). The Gamma prior becomes non-informative in the limit $a\to 0$, $b\to 0$. Since, in this limit, the hyperpriors become improper, we might fix their parameters to small values: *e.g.* $a = b = c = d = 10^{-4}$. However, by setting these parameters to zero, we obtain uniform hyperpriors (over a *logarithmic* scale). Since all scales are equally likely, a pleasing consequence of the use of these improper hyperpriors is that of scale-invariance: predictions are independent of linear scaling of both $\mathbf{t}$ and the basis function outputs so, for example, results do not depend on the unit of measurement of the targets. The case of general Gamma priors for $\boldsymbol{\alpha}$ and β is covered in more detail in Tipping (2001) and Bishop and Tipping (2000), but from now on here we assume uniform scale priors with $a = b = c = d = 0$.

This choice of prior distributions is related to those used in *automatic relevance determination*, or ARD (MacKay 1994; Neal 1996). Using such priors in a neural network, individual hyperparameters would typically control *groups* of weights, in particular those associated

with each input dimension x. For inputs which have little value in predicting the outputs, the posterior distribution over the hyperparameters becomes concentrated at large values, thus effectively switching off such 'low relevance' inputs. This idea has also been applied to the input variables in 'Gaussian process' models (Williams 1997).

Here, the assignment of an individual hyperparameter to each weight, or basis function, is the key feature of the sparse Bayesian framework, and is responsible ultimately for its sparsity properties. To introduce an additional $N + 1$ parameters to the model may seem counter-intuitive, since there is already one parameter per basis function (and therefore one parameter per data point for kernel functions centered on the data), but from a Bayesian perspective, provided we correctly integrate out all of these parameters, or can approximate such an integration sufficiently accurately, then having the number of parameters exceed the number of data points presents no particular difficulty either from a theoretical or from a practical point of view (see pp. 16–17, of Neal (1996)).

3.2 The Effective Prior

We may question why the choice of a Gaussian prior should express any preference for sparse models. In order to gain insight into the this effect we can integrate out the hyperparameters to discover the true identity of the prior over the weights. For a Gamma prior over the hyperparameters, it is possible to integrate out α, independently for each weight, to obtain the marginal, or what might be considered the 'true', weight prior:

$$\begin{aligned} p(w_i) &= \int p(w_i|\alpha_i)p(\alpha_i)\, d\alpha_i, \\ &= \frac{b^a \Gamma(a + \frac{1}{2})}{(2\pi)^{\frac{1}{2}}\Gamma(a)} \, (b + w_i^2/2)^{-\left(a+\frac{1}{2}\right)}, \end{aligned} \tag{22}$$

where $\Gamma(\cdot)$ is the gamma function as defined earlier. Equation (22) corresponds to the density of a Student-t distribution, and so the overall marginal weight prior is a product of independent Student-t distributions over the w_i. A visualization of this Student-t prior, alongside a Gaussian, is given in Figure 1. For the case of the uniform hyperprior, with $a = b = 0$, we obtain the improper prior $p(w_i) \propto 1/|w_i|$. Intuitively, this looks very much like a sparse prior since it is sharply peaked at zero like the popular Laplace prior $p(w_i) \propto \exp(-|w_i|)$, which has been previously utilized to obtain sparsity in Bayesian contexts (Williams 1995). The elegance of this approach therefore lies in the use of hierarchical modelling to obtain a prior over weights which encourages sparsity while still making use of fully conjugate exponential-family distributions throughout.

Unfortunately, we cannot continue the Bayesian analysis down this route to compute $p(\mathbf{w}|\mathbf{t})$, since the marginal $p(\mathbf{w})$ is no longer Gaussian, and so the marginalization over $\mathbf{w}$ is no longer analytically tractable. Because of this, it is not convenient to work with the marginal prior directly and in the next section we take a different tack.

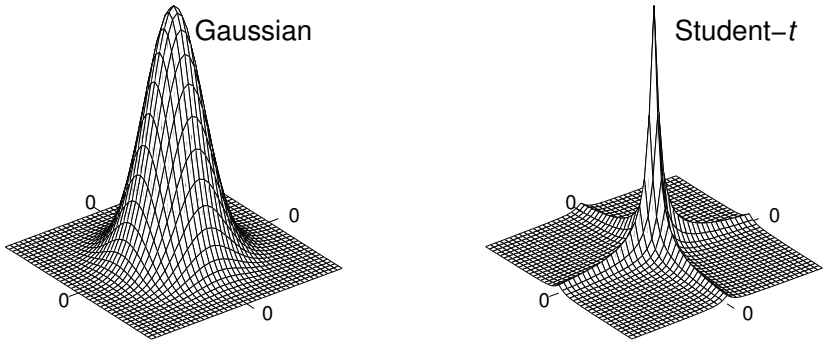

Figure 1: LEFT: an example Gaussian prior $p(\mathbf{w}|\boldsymbol{\alpha})$ in two dimensions. RIGHT: the prior $p(\mathbf{w})$, where the hyperparameters have been integrated out to give a product of Student-t distributions. Note that the probability mass is concentrated close to the origin, where both weights go to zero, and also along 'spines' where one or other of the two weights goes to zero.

3.3 Inference

Having defined the prior, Bayesian inference proceeds by computing, from Bayes' rule, the posterior over all unknowns given the data:

$$p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2|\mathbf{t}) = \frac{p(\mathbf{t}|\mathbf{w}, \boldsymbol{\alpha}, \sigma^2)p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2)}{p(\mathbf{t})}. \tag{23}$$

Then, given a new test point, $\mathbf{x}_*$, predictions are made for the corresponding target t_*, in terms of the predictive distribution:

$$p(t_*|\mathbf{t}) = \int p(t_*|\mathbf{w}, \boldsymbol{\alpha}, \sigma^2)\; p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2|\mathbf{t})\; d\mathbf{w}\; d\boldsymbol{\alpha}\; d\sigma^2. \tag{24}$$

As is the case with many non-trivial Bayesian models, it is not possible to perform these computations in full analytically, and we must seek an effective approximation.

We cannot compute the posterior $p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2|\mathbf{t})$ in (23) directly since we cannot perform the normalizing integral on the right-hand-side, $p(\mathbf{t}) = \int p(\mathbf{t}|\mathbf{w}, \boldsymbol{\alpha}, \sigma^2)p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2)\; d\mathbf{w}\; d\boldsymbol{\alpha}\; d\sigma^2$. Instead, we decompose the posterior as:

$$p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2|\mathbf{t}) = p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha}, \sigma^2)p(\boldsymbol{\alpha}, \sigma^2|\mathbf{t}), \tag{25}$$

and note that we can compute analytically the posterior distribution over the weights since its normalizing integral, $p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2) = \int p(\mathbf{t}|\mathbf{w}, \sigma^2)\; p(\mathbf{w}|\boldsymbol{\alpha})\; d\mathbf{w}$, is a convolution of Gaussians. The posterior distribution over the weights is thus given by:

$$p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha}, \sigma^2) = \frac{p(\mathbf{t}|\mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha})}{p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2)}, \tag{26}$$

$$= (2\pi)^{-(N+1)/2}|\boldsymbol{\Sigma}|^{-1/2}\; \exp\left\{-\frac{1}{2}(\mathbf{w} - \boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{w} - \boldsymbol{\mu})\right\}, \tag{27}$$

where the posterior covariance and mean are respectively:

$$\boldsymbol{\Sigma} = (\sigma^{-2}\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi} + \mathbf{A})^{-1}, \tag{28}$$

$$\boldsymbol{\mu} = \sigma^{-2}\boldsymbol{\Sigma}\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t}, \tag{29}$$

with $\mathbf{A} = \mathrm{diag}(\alpha_0, \alpha_1, \ldots, \alpha_N)$.

It is now necessary to make some form of approximation, and we do so by replacing the integration over the hyperparameters by point estimates involving their most probable posterior values. We do this on the basis that this point-estimate is representative of the posterior in the sense that functions generated utilizing the posterior mode values are close to those obtained by sampling from the full posterior distribution. It is important to realize that this does not necessitate that the entire mass of the posterior be accurately approximated by the delta-function. For predictive purposes, rather than requiring $p(\boldsymbol{\alpha}, \sigma^2|\mathbf{t}) \approx \delta(\boldsymbol{\alpha}_{\mathrm{MP}}, \sigma^2_{\mathrm{MP}})$, we only desire

$$\int p(t_*|\boldsymbol{\alpha}, \sigma^2) p(\boldsymbol{\alpha}, \sigma^2|\mathbf{t})\; d\boldsymbol{\alpha}\; d\sigma^2 \simeq p(t_*|\boldsymbol{\alpha}_{\mathrm{MP}}, \sigma^2_{\mathrm{MP}}) \tag{30}$$

to be a good approximation. This notion may be visualized by a thought experiment where we consider that we are utilizing two identical basis functions $\phi_i(\mathbf{x})$ and $\phi_j(\mathbf{x})$. It follows from (31) shortly that the mode of $p(\boldsymbol{\alpha}, \sigma^2|\mathbf{t})$ will not be unique, but will comprise an infinite 'ridge' where $\alpha_i^{-1} + \alpha_j^{-1}$ is some constant value. No delta-function can be considered to be a good approximation to the probability mass associated with this ridge, yet any point along it implies an identical predictive distribution and so (30) holds. Evidence from the experiments presented in this article and elsewhere suggests that this predictive approximation is very effective in general.

Relevance vector 'learning' thus becomes the search for the hyperparameter posterior mode, *i.e.* the maximization of $p(\boldsymbol{\alpha}, \sigma^2|\mathbf{t}) \propto p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2) p(\boldsymbol{\alpha}) p(\sigma^2)$ with respect to $\boldsymbol{\alpha}$ and β. For the case of uniform hyperpriors, we need only maximize the *marginal likelihood*, or equivalently its logarithm, $\ln p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2)$, which is computable and given by:

$$\begin{aligned} \mathcal{L}(\boldsymbol{\alpha}) = \ln p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2) &= \ln \int_{-\infty}^{\infty} p(\mathbf{t}|\mathbf{w}, \sigma^2)\; p(\mathbf{w}|\boldsymbol{\alpha})\; d\mathbf{w}, \\ &= -\frac{1}{2}\left[N \ln 2\pi + \ln|\mathbf{C}| + \mathbf{t}^{\mathrm{T}}\mathbf{C}^{-1}\mathbf{t}\right], \end{aligned} \tag{31}$$

with

$$\mathbf{C} = \sigma^2\mathbf{I} + \boldsymbol{\Phi}\mathbf{A}^{-1}\boldsymbol{\Phi}^{\mathrm{T}}. \tag{32}$$

3.4 Making Predictions

In practice, having maximized (31) (we consider this task shortly), we make predictions based on the posterior distribution over the weights, conditioned on the maximizing values $\boldsymbol{\alpha}_{\mathrm{MP}}$ and σ^2_{MP}. We can then compute the predictive distribution, from (24), for a new datum $\mathbf{x}_*$ using (27):

$$p(t_*|\mathbf{t}, \boldsymbol{\alpha}_{\mathrm{MP}}, \sigma^2_{\mathrm{MP}}) = \int p(t_*|\mathbf{w}, \sigma^2_{\mathrm{MP}}) p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha}_{\mathrm{MP}}, \sigma^2_{\mathrm{MP}})\; d\mathbf{w}. \tag{33}$$

Since both terms in the integrand are Gaussian, this is readily computed, giving:

$$p(t_*|\mathbf{t}, \boldsymbol{\alpha}_{\mathrm{MP}}, \sigma^2_{\mathrm{MP}}) = \mathcal{N}(t_*|y_*, \sigma^2_*),$$

with

$$y_* = \boldsymbol{\mu}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_*), \tag{34}$$

$$\sigma^2_* = \sigma^2_{\mathrm{MP}} + \boldsymbol{\phi}(\mathbf{x}_*)^{\mathrm{T}}\boldsymbol{\Sigma}\boldsymbol{\phi}(\mathbf{x}_*). \tag{35}$$

So the predictive mean is intuitively $y(\mathbf{x}_*; \boldsymbol{\mu})$, or the basis functions weighted by the posterior mean weights. We will find that the maximizing values of many of the hyperparameters will be infinite, implying from (27) that the corresponding weights in $\mathbf{w}_{\mathrm{MP}}$ will be exactly zero and the predictor y_* is thus sparse.

3.5 Properties of the Marginal Likelihood

Values of $\boldsymbol{\alpha}$ (assume σ^2 is fixed for now) which maximize (31) cannot be jointly obtained in closed form. However, in Faul and Tipping (2002) it was shown that we can maximize $\mathcal{L}(\boldsymbol{\alpha})$ with respect to a *single* hyperparameter α_i. To show this, we first straightforwardly decompose $\mathbf{C}$ in (32) as

$$\begin{aligned}\mathbf{C} &= \sigma^2\mathbf{I} + \sum_{m\neq i} \alpha_m^{-1}\boldsymbol{\phi}_m\boldsymbol{\phi}_m^{\mathrm{T}} + \alpha_i^{-1}\boldsymbol{\phi}_i\boldsymbol{\phi}_i^{\mathrm{T}},\\ &= \mathbf{C}_{-i} + \alpha_i^{-1}\boldsymbol{\phi}_i\boldsymbol{\phi}_i^{\mathrm{T}},\end{aligned} \tag{36}$$

where $\mathbf{C}_{-i}$ is $\mathbf{C}$ with the contribution of basis vector i removed and $\boldsymbol{\phi}_i$ is the i-th column of $\boldsymbol{\Phi}$. Established matrix determinant and inverse identities can then be employed to re-write $\mathcal{L}(\boldsymbol{\alpha})$ as:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\alpha}) &= -\frac{1}{2}\Big[N\ln(2\pi) + \ln|\mathbf{C}_{-i}| + \mathbf{t}^{\mathrm{T}}\mathbf{C}_{-i}^{-1}\mathbf{t}\\ &\qquad - \ln\alpha_i + \ln(\alpha_i + \boldsymbol{\phi}_i^{\mathrm{T}}\mathbf{C}_{-i}^{-1}\boldsymbol{\phi}_i) - \frac{(\boldsymbol{\phi}_i^{\mathrm{T}}\mathbf{C}_{-i}^{-1}\mathbf{t})^2}{\alpha_i + \boldsymbol{\phi}_i^{\mathrm{T}}\mathbf{C}_{-i}^{-1}\boldsymbol{\phi}_i}\Big],\\ &= \mathcal{L}(\boldsymbol{\alpha}_{-i}) + \frac{1}{2}\left[\ln\alpha_i - \ln(\alpha_i + s_i) + \frac{q_i^2}{\alpha_i + s_i}\right]\\ &= \mathcal{L}(\boldsymbol{\alpha}_{-i}) + \ell(\alpha_i),\end{aligned} \tag{37}$$

where for simplification of forthcoming expressions, we have defined:

$$s_i \triangleq \boldsymbol{\phi}_i^{\mathrm{T}}\mathbf{C}_{-i}^{-1}\boldsymbol{\phi}_i, \quad \text{and} \quad q_i \triangleq \boldsymbol{\phi}_i^{\mathrm{T}}\mathbf{C}_{-i}^{-1}\mathbf{t}. \tag{38}$$

The objective function has now been decomposed into $\mathcal{L}(\boldsymbol{\alpha}_{-i})$, the marginal likelihood with $\boldsymbol{\phi}_i$ excluded, and $\ell(\alpha_i)$, where terms in α_i are now conveniently isolated.

Analysis of $\ell(\alpha_i)$ (Faul and Tipping 2002) shows that $\mathcal{L}(\boldsymbol{\alpha})$ has a unique maximum with respect to α_i:

$$\alpha_i = \frac{s_i^2}{q_i^2 - s_i}, \qquad \text{if } q_i^2 > s_i, \tag{39}$$

$$\alpha_i = \infty, \qquad \text{if } q_i^2 \leq s_i. \tag{40}$$

An example illustrating these two cases is given in Figure 2.

Thus sparsity arises from equation (40): if this condition holds, then the marginal likelihood is maximized when the individual basis (kernel) function $K(\mathbf{x}, \mathbf{x}_i)$ is removed from the model. Note, from (38), that q_i and s_i depend on all other ($m \neq i$) hyperparameters so the sparsity conditions for all $\boldsymbol{\alpha}$ mutually interact.

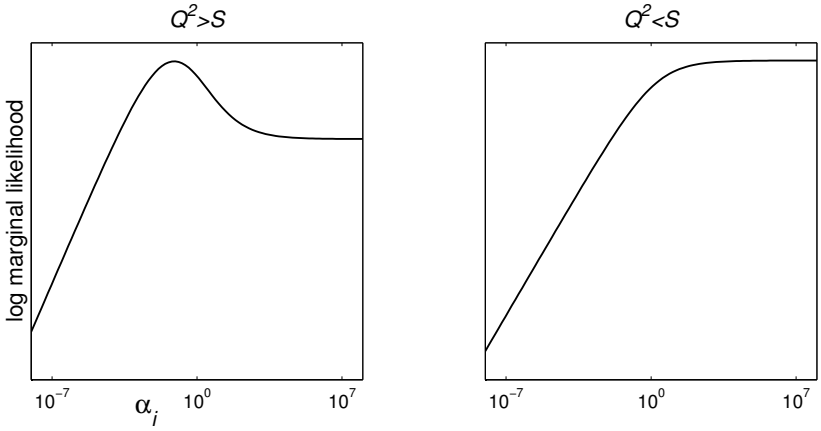

Figure 2: Example plots of $\ell(\alpha_i)$ against α_i (on a log scale) for $q^2 > s$ (left), showing the single maximum at finite α_i, and $q^2 < s$ (right), showing the maximum as $\alpha_i \to \infty$.

3.6 Hyperparameter Optimization

For optimizing hyperparameters, a simple set of re-estimation formulae can be derived (Tipping 2000; Tipping 2001), but a more recent, and much more efficient, approach is given in Tipping and Faul (2003) which we briefly summarize here.

We start by computing the quantities s_i and q_i. In fact, it is easier to maintain and update values of

$$S_i = \boldsymbol{\phi}_i^{\mathrm{T}} \mathbf{C}^{-1} \boldsymbol{\phi}_i, \qquad Q_i = \boldsymbol{\phi}_i^{\mathrm{T}} \mathbf{C}^{-1} \mathbf{t}, \tag{41}$$

and from these it follows simply:

$$s_i = \frac{\alpha_i S_i}{\alpha_i - S_i}, \qquad q_i = \frac{\alpha_i Q_i}{\alpha_i - S_i}. \tag{42}$$

Note that when $\alpha_i = \infty$, $s_i = S_i$ and $q_i = Q_i$. In practice, then, it is convenient to utilise the Woodbury identity to obtain the quantities of interest:

$$S_i = \boldsymbol{\phi}_i^{\mathrm{T}} \mathbf{B} \boldsymbol{\phi}_i - \boldsymbol{\phi}_i^{\mathrm{T}} \mathbf{B} \boldsymbol{\Phi} \widehat{\boldsymbol{\Sigma}} \boldsymbol{\Phi}^{\mathrm{T}} \mathbf{B} \boldsymbol{\phi}_i, \tag{43}$$

$$Q_i = \boldsymbol{\phi}_i^{\mathrm{T}} \mathbf{B} \widehat{\mathbf{t}} - \boldsymbol{\phi}_i^{\mathrm{T}} \mathbf{B} \widehat{\boldsymbol{\mu}}, \tag{44}$$

where $\mathbf{B} \equiv \sigma^{-2}\mathbf{I}$, $\widehat{\boldsymbol{\Sigma}} \equiv \boldsymbol{\Sigma}$, $\widehat{\boldsymbol{\mu}} \equiv \boldsymbol{\mu}$ and $\widehat{\mathbf{t}} \equiv \mathbf{t}$ in the regression case, and for the classification case as explicitly defined in the next section.

Given these, consider individual basis functions (hyperparameters) in turn and note that the results (39) and (40) imply that for a given α_i:

- If $\boldsymbol{\phi}_i$ is 'in the model' (*i.e.* $\alpha_i < \infty$) yet $q_i^2 \leq s_i$, then $\boldsymbol{\phi}_i$ may be deleted (*i.e.* α_i set to ∞),
- If $\boldsymbol{\phi}_i$ is excluded from the model ($\alpha_i = \infty$) and $q_i^2 > s_i$, $\boldsymbol{\phi}_i$ may be 'added': *i.e.* α_i is set to the optimal finite value given by (39).
- If $\boldsymbol{\phi}_i$ is 'in the model' and $q_i^2 > s_i$ then α_i may be re-estimated.

All these actions guarantee to increase the marginal likelihood function, and we thus have a framework for making discrete changes to the model, by adding and deleting basis functions, in a principled probabilistic manner.

For the noise variance σ^2, we can derive a re-estimation equation which may be utilized concurrently with those of $\boldsymbol{\alpha}$ in order to infer the noise variance:

$$(\sigma^2)^{\text{new}} = \frac{\|\mathbf{t} - \boldsymbol{\Phi}\boldsymbol{\mu}\|^2}{N - M + \sum_i \alpha_i \Sigma_{ii}}. \tag{45}$$

Note that the 'N' in the denominator refers to the number of data examples and not the number of basis functions.

Further information on the optimization procedure, including details of computing and updating s_i and q_i are given in Tipping and Faul (2003).

3.7 Relevance Vector Machines for Classification

Sparse Bayesian classification follows an essentially identical framework as described for regression above, but using a Bernoulli likelihood and a sigmoidal link function to account for the change in the target quantities. As a consequence, there is an additional approximation step in the algorithm.

Applying the logistic sigmoid link function $\sigma(y) = 1/(1 + e^{-y})$ to $y(\mathbf{x}; \mathbf{w})$ and, adopting the Bernoulli distribution for $P(t|\mathbf{x})$, we write the likelihood as:

$$P(\mathbf{t}|\mathbf{w}) = \prod_{n=1}^{N} \sigma\{y(\mathbf{x}_n; \mathbf{w})\}^{t_n} \left[1 - \sigma\{y(\mathbf{x}_n; \mathbf{w})\}\right]^{1-t_n}, \tag{46}$$

where, following from the probabilistic specification, the targets $t_n \in \{0, 1\}$.

Unlike the regression case, the weights cannot be integrated out analytically, precluding closed-form expressions for either the weight posterior $p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha})$ or the marginal likelihood $P(\mathbf{t}|\boldsymbol{\alpha})$. We thus utilize the Laplace approximation procedure, as used in MacKay (1992):

1. For the current values of $\boldsymbol{\alpha}$, the the mode of the posterior distribution is found iteratively to give the 'most probable' weights $\widehat{\boldsymbol{\mu}}$. Since $p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha}) \propto P(\mathbf{t}|\mathbf{w})p(\mathbf{w}|\boldsymbol{\alpha})$, this is equivalent to finding the maximum, over $\mathbf{w}$, of

$$\ln \{P(\mathbf{t}|\mathbf{w})p(\mathbf{w}|\boldsymbol{\alpha})\} = \sum_{n=1}^{N} [t_n \ln y_n + (1 - t_n) \ln(1 - y_n)] - \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{A}\mathbf{w}, \tag{47}$$

   with $y_n = \sigma\{y(\mathbf{x}_n; \mathbf{w})\}$. This is a standard procedure, since (47) is a penalized logistic log-likelihood function, and necessitates iterative maximization. We have used a second-order Newton method related to the 'iteratively-reweighted least-squares' algorithm to find $\widehat{\boldsymbol{\mu}}$.

2. Laplace's method is simply a quadratic approximation to the log-posterior around its mode. The quantity (47) is differentiated twice to give:

$$\nabla_{\mathbf{w}} \nabla_{\mathbf{w}} \ln p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha})\big|_{\widehat{\boldsymbol{\mu}}} = -(\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{B}\boldsymbol{\Phi} + \mathbf{A}), \tag{48}$$

   where $\mathbf{B} = \text{diag}\,(\beta_1, \beta_2, \ldots, \beta_N)$ is a diagonal matrix with $\beta_n = \sigma\{y(\mathbf{x}_n)\}\left[1 - \sigma\{y(\mathbf{x}_n)\}\right]$. This is then negated and inverted to give the covariance $\widehat{\boldsymbol{\Sigma}}$ for a Gaussian approximation to the posterior over weights centered at $\widehat{\boldsymbol{\mu}}$.

At the mode of $p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha})$, using (48) and the fact that $\nabla_{\mathbf{w}} \ln p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha})\big|_{\widehat{\boldsymbol{\mu}}} = 0$, we can see we have effectively locally 'linearized' the classification problem around $\widehat{\boldsymbol{\mu}}$ with

$$\widehat{\boldsymbol{\Sigma}} = (\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{B}\boldsymbol{\Phi} + \mathbf{A})^{-1}, \tag{49}$$

$$\widehat{\boldsymbol{\mu}} = \widehat{\boldsymbol{\Sigma}}\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{B}\widehat{\mathbf{t}}, \tag{50}$$

and

$$\widehat{\mathbf{t}} = \boldsymbol{\Phi}\widehat{\boldsymbol{\mu}} + \mathbf{B}^{-1}(\mathbf{t} - \sigma\{\boldsymbol{\Phi}\widehat{\boldsymbol{\mu}}\}). \tag{51}$$

These equations are equivalent to the solution to a generalized least squares problem. Compared with (29), it can be seen that the Laplace approximation effectively maps the classification problem to a regression one with targets $\widehat{\mathbf{t}}$ and data-dependent (heteroscedastic) noise, in which the inverse noise variance for ϵ_n is given by $\beta_n = \sigma\{y(\mathbf{x}_n)\}\,[1 - \sigma\{y(\mathbf{x}_n)\}]$.

The quantities $\widehat{\boldsymbol{\Sigma}}$, $\widehat{\boldsymbol{\mu}}$ and $\widehat{\mathbf{t}}$ can be substituted into equations (43) and (44) in order to compute the quantities s_i and q_i which may then be exploited in the algorithm of Section 3.6 exactly as in the regression case.

4 The Relevance Vector Machine in Action

4.1 Illustrative synthetic data: regression

The function $\mathrm{sinc}(x) = \sin(x)/x$ has been a popular choice to illustrate support vector regression (Vapnik, Golowich, and Smola 1997; Vapnik 1998), where in place of the classification margin, the *ϵ-insensitive region* is introduced, a 'tube' of $\pm\epsilon$ around the function within which errors are not penalized. In this case, the support vectors lie on the edge of, or outside, this region. For example, using a univariate 'linear spline' kernel:

$$K(x_m, x_n) = 1 + x_m x_n + x_m x_n \min(x_m, x_n) - \frac{x_m + x_n}{2}\min(x_m, x_n)^2 + \frac{\min(x_m, x_n)^3}{3}, \tag{52}$$

and with $\epsilon = 0.01$, the approximation of $\mathrm{sinc}(x)$ based on 100 uniformly-spaced noise-free samples in $[-10, 10]$ utilizes 36 support vectors as shown in Figure 3 (left).

In the RVM, we model the same data with the same kernel (52), which is utilized to define a set of basis functions $\phi_n(x) = K(x, x_n)$, $n = 1 \ldots N$. Typically, we will be tackling problems where the target function has some additive noise component, whose variance is represented by σ^2. However, for the purposes of comparison with this "function approximation" SVM example, we model the sinc function with a relevance vector machine but *fix* the noise variance in this case at 0.01^2 and then re-estimate $\boldsymbol{\alpha}$ alone. This setting of the noise standard deviation to 0.01 is intended to be analogous, in an approximate sense, to the setting the ϵ-insensitivity to the same value in the SVM. Using this fixed σ, the RVM approximator is plotted in Figure 3 (right), and requires only 9 relevance vectors. The largest error is 0.0070, compared to 0.010 in the support vector case, and we have obtained the dual benefit of both increased accuracy and sparsity.

Figure 4 illustrates a case which is more representative of real data in which uniform noise (*i.e.* not corresponding to the RVM noise model) in $[-0.2, 0.2]$ is added to the targets. Again, a linear spline kernel was used. The trained RVM uses 6 relevance vectors, compared to 29 for the SVM. The root-mean-square (RMS) deviation from the true function for the RVM is 0.0245, while for the SVM it is 0.0291. Note that for the latter model, it was necessary

to tune the parameters C and ϵ, in this case using 5-fold cross-validation. For the RVM, the analogues of these parameters (the α's and σ^2) are automatically estimated by the learning procedure.

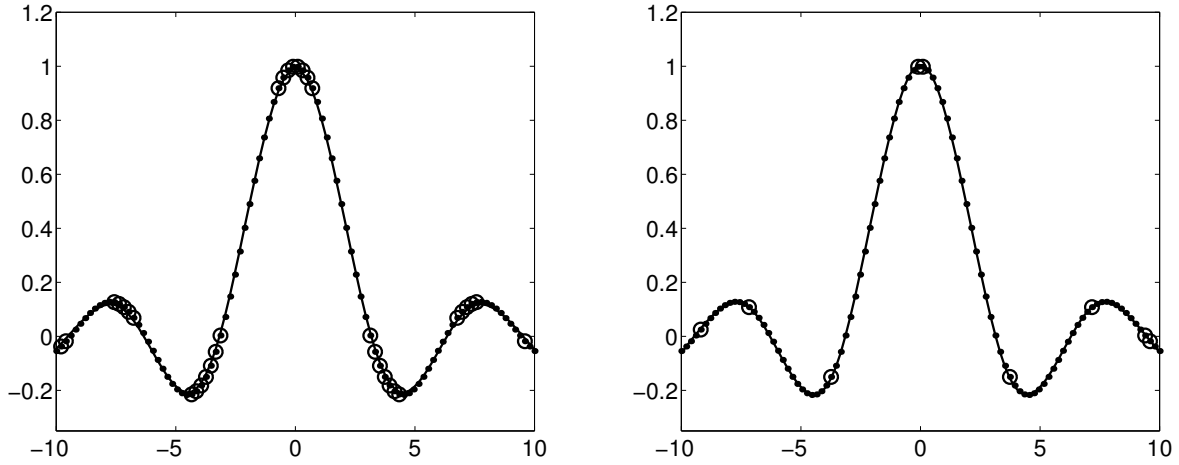

Figure 3: Support (left) and relevance (right) vector approximations to $\mathrm{sinc}(x)$ from 100 noise-free examples using 'linear spline' basis functions. The estimated functions are drawn as solid lines with support/relevance vectors shown circled.

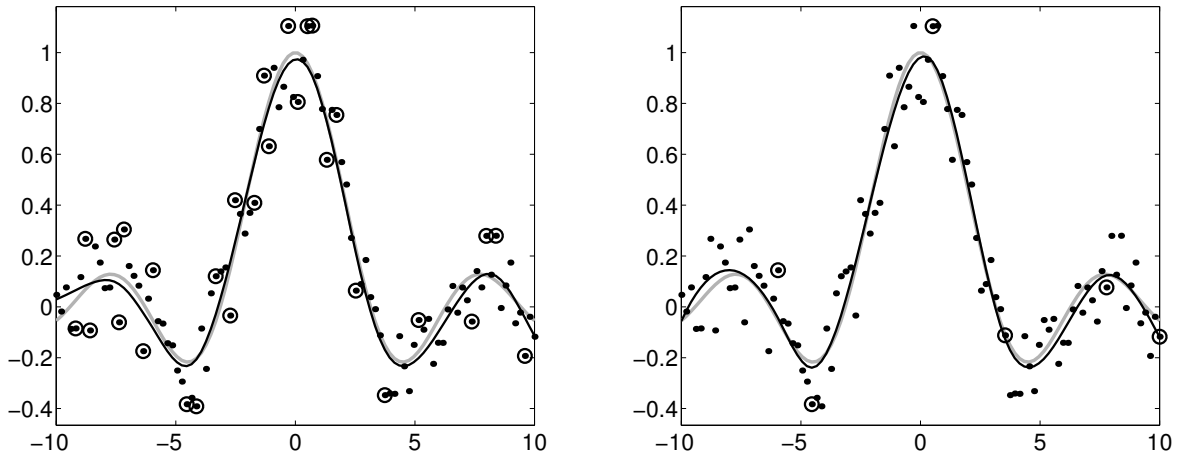

Figure 4: Support (left) and relevance (right) vector approximations to $\mathrm{sinc}(x)$, based on 100 noisy samples. The estimated functions are drawn as solid lines, the true function in grey, and support/relevance vectors are again shown circled.

4.2 Illustrative synthetic data: classification

We utilize artificially-generated data in two dimensions in order to illustrate graphically the selection of relevance vectors for classification. Both class 1 (denoted by '$\times$') and class 2 (denoted by '$\bullet$') were generated from mixtures of two Gaussians by Ripley (1996), with the classes overlapping to the extent that the Bayes error is around 8%.

A relevance vector classifier is compared to its support vector counterpart, using a 'Gaussian' kernel which we define as

$$K(\mathbf{x}_m, \mathbf{x}_n) = \exp(-r^{-2}\|\mathbf{x}_m - \mathbf{x}_n\|^2), \tag{53}$$

with r the 'width' parameter, chosen here to be 0.5. A value of C for the SVM was selected using 5-fold cross-validation on the training set. The results for a 100-example training set (randomly chosen from Ripley's original 250) are given in Figure 5. The test error (from the associated 1000-example test set) for the RVM (9.3%) is slightly superior to the SVM (10.6%), but the remarkable feature of contrast is the complexity of the classifiers. The support vector machine utilizes 38 kernel functions compared to just 4 for the relevance vector method. This considerable difference in sparsity between the two methods is typical, as the later results on benchmark data sets support.

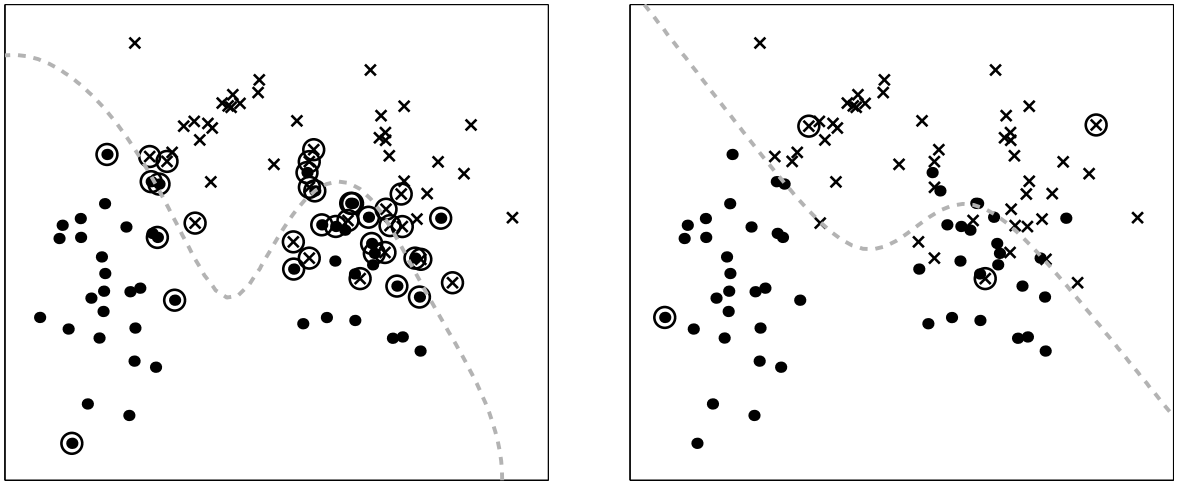

Figure 5: SVM (left) and RVM (right) classifiers on 100 examples from Ripley's Gaussian-mixture data set. The decision boundary is shown dashed, and relevance/support vectors are shown circled to emphasize the dramatic reduction in complexity of the RVM model.

Of interest also is the fact that, unlike with the SVM, the relevance vectors are some distance from the decision boundary (in $\mathbf{x}$-space), appearing more 'prototypical' or even 'anti-boundary' in character. A qualitative explanation for this phenomenon, discussed in more detail in Tipping (2001), is that the output of a basis function centered on or near the decision boundary is an unreliable indicator of class membership (*i.e.* its output is poorly-aligned with the data set in $\mathbf{t}$-space), and such basis functions are naturally penalized (deemed 'irrelevant') under the Bayesian framework. Of course, there is no implication that the utilization of either boundary-located or prototypically-located functions is 'correct' in any sense.

4.3 Benchmark Results

The following tables, taken from Tipping (2001), summarize regression and classification performance of the relevance vector machine on some example benchmark data sets, comparing results for illustrative purposes with equivalent support vector machines. For each data set the number of training examples (N) and the number of input variables (d) are given in the tables. The prediction error obtained and the number of vectors (support or relevance) required, generally averaged over a number of repetitions, are then given for both models. By way of summary, the RVM statistics were also normalized by those of the SVM and the overall average is displayed. A Gaussian kernel was utilized and its input scale parameter chosen by 5-fold cross-validation.

			errors		*vectors*	
Regression Data set	***N***	***d***	**SVM**	**RVM**	**SVM**	**RVM**
Sinc (Gaussian noise)	100	1	0.0378	0.0326	45.2	6.7
Sinc (Uniform noise)	100	1	0.0215	0.0187	44.3	7.0
Friedman #2	240	4	4140	3505	110.3	6.9
Friedman #3	240	4	0.0202	0.0164	106.5	11.5
Boston Housing	481	13	8.04	7.46	142.8	39.0
Normalized Mean			1.00	**0.86**	1.00	**0.15**

			errors		*vectors*	
Classification Data set	***N***	***d***	**SVM**	**RVM**	**SVM**	**RVM**
Pima Diabetes	200	8	20.1%	19.6%	109	4
U.S.P.S.	7291	256	4.4%	5.1%	2540	316
Banana	400	2	10.9%	10.8%	135.2	11.4
Breast Cancer	200	9	26.9%	29.9%	116.7	6.3
Titanic	150	3	22.1%	23.0%	93.7	65.3
Waveform	400	21	10.3%	10.9%	146.4	14.6
German	700	20	22.6%	22.2%	411.2	12.5
Image	1300	18	3.0%	3.9 %	166.6	34.6
Normalized Mean			1.00	**1.08**	1.00	**0.17**

In summary, in this small number of experiments, the RVM exhibited 14% lower error than the SVM and utilized only 15% of the basis functions on average for regression. In classification, error was 8% greater on average, yet still only 17% of the basis functions were utilized.

5 Discussion

In this brief tutorial we have outlined some of the basic concepts of regression and classification from the Bayesian perspective. We have discussed in detail a specific Bayesian model called the Relevance Vector Machine, which leads to highly sparse solutions and having excellent generalization properties.

The treatment of the Relevance Vector Machine given here is not completely Bayesian since point estimates are made for the hyperparameters, whereas in a fully Bayesian treatment we should define hyperpriors over these hyperparameters, and then integrate out the hyperparameters in order to make predictions.

However, as we have already noted, it is not possible to integrate out all of the parameters and hyperparameters analytically. This problem can be addressed by using deterministic approximation schemes based *variational inference*, in which a factorized approximation to the full posterior distribution is used (Bishop and Tipping 2000). One consequence of this more complete treatment of the RVM is confirmation that the approach based on point estimates, as discussed in this tutorial, does indeed give a good approximation to a more complete Bayesian approach.

References

Abromowitz, M. and I. A. Stegun (1965). *Handbook of Mathematical Functions*. Dover.

Berger, J. O. (1985). *Statistical decision theory and Bayesian analysis* (Second ed.). Springer.

Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press.

Bishop, C. M. and M. E. Tipping (2000). Variational relevance vector machines. In C. Boutilier and M. Goldszmidt (Eds.), *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pp. 46–53. Morgan Kaufmann.

Boser, B., I. Guyon, and V. N. Vapnik (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, pp. 144–152.

Faul, A. C. and M. E. Tipping (2002). Analysis of sparse Bayesian learning. In T. G. Dietterich, S. Becker, and Z. Ghahramani (Eds.), *Advances in Neural Information Processing Systems 14*, pp. 383–389. MIT Press.

MacKay, D. J. C. (1992). The evidence framework applied to classification networks. *Neural Computation 4*(5), 720–736.

MacKay, D. J. C. (1994). Bayesian methods for backpropagation networks. In E. Domany, J. L. van Hemmen, and K. Schulten (Eds.), *Models of Neural Networks III*, Chapter 6, pp. 211–254. Springer.

Neal, R. M. (1996). *Bayesian Learning for Neural Networks*. Springer.

Ripley, B. D. (1996). *Pattern Recognition and Neural Networks*. Cambridge University Press.

Schölkopf, B., C. J. C. Burges, and A. J. Smola (Eds.) (1999). *Advances in Kernel Methods: Support Vector Learning*. MIT Press.

Tipping, M. E. (2000). The Relevance Vector Machine. In S. A. Solla, T. K. Leen, and K.-R. Müller (Eds.), *Advances in Neural Information Processing Systems 12*, pp. 652–658. MIT Press.

Tipping, M. E. (2001). Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research 1*, 211–244.

Tipping, M. E. and A. C. Faul (2003). Fast marginal likelihood maximisation for sparse Bayesian models. In B. Frey and C. M. Bishop (Eds.), *Artificial Intelligence and Statistics*. To appear.

Vapnik, V. N. (1998). *Statistical Learning Theory*. Wiley.

Vapnik, V. N., S. E. Golowich, and A. J. Smola (1997). Support vector method for function approximation, regression estimation and signal processing. In M. C. Mozer, M. I. Jordan, and T. Petsche (Eds.), *Advances in Neural Information Processing Systems 9*. MIT Press.

Williams, C. K. I. (1997). Prediction with gaussian processes: From linear regression to linear prediction and beyond. Technical report, NCRG, Aston University, Birmingham, U.K.

Williams, P. M. (1995). Bayesian regularisation and pruning using a Laplace prior. *Neural Computation 7*(1), 117–143.